

**UNIVERSITY of  
STIRLING**



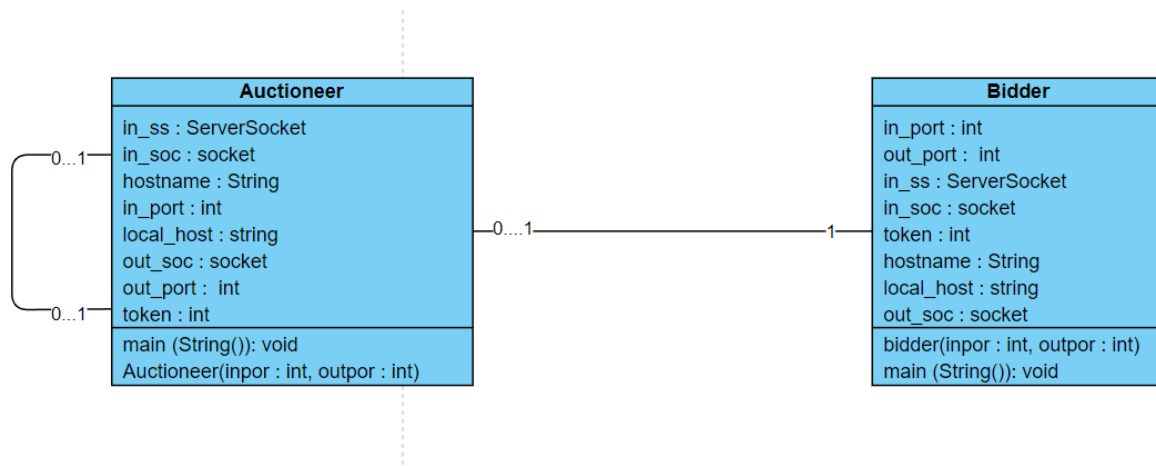
**Student Number: 3058058**

**Title: CSCU9V5 Distributed Systems ASSIGNMENT**

**Module: CSCU9V5 (Core) - Operating Systems Concurrency  
and Distribution**

## Report:

### Class Diagram with relevant Relationships:



### Basic Solution:

#### Auctioneer Class:

To begin with the `_bid` was initialized at 100. After that the `bid.txt` file was needed to store the initial bid, which was easily completed. The `current_bid.seek(0)` was placed there because I wanted to make sure the bid was written down properly on the text file but isn't necessary. After that, we had to make sure the token left from the auctioneer out socket and goes to the local host in socket. Firstly, I had to create a new socket for the out socket of the auctioneer named `out_soc`. So, I had to first make sure that the socket identified the local host and the `out_port` (the `in_port` of the bidder) for the auctioneer and create the out socket itself. Then I had to make sure that the token was printed and flushed into the stream for the bidders to receive and then the socket was closed. After that we had to create a server socket called `in_ss` another socket for the auctioneer called the in socket to receive the token.

#### Bidder class:

In the bidder class we had to make first called in `in_ss` for the socket for the `in_port` called `in_soc`. The socket is open, and a random number is generated. If the number generation happens then the new value is written and is overridden in the `_bid.txt` (which is updated with the new bid). The out socket is the same as in auctioneer and sends the token to the next in socket of the next bidder.

#### Run Configurations:

Three Bidders and a single auctioneer configuration is made to run the program with the only parameters of the out sockets and in sockets given in the arguments.

## Auctioneer basic solution output:

```
<terminated> Auctioneer [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe (Dec 5, 2021, 11:02:34 AM – 11:02:39 AM)
node hostname is LAPTOP-OBTQ2TF0:LAPTOP-OBTQ2TF0/127.0.0.1
Auctioneer: 7000 of distributed lottery is active ....
Auctioneer: 7000 - STARTING AUCTION with price = 100
Auctioneer: 7000 :: sent token to 7001
Auctioneer: 7000 :: received token back
```

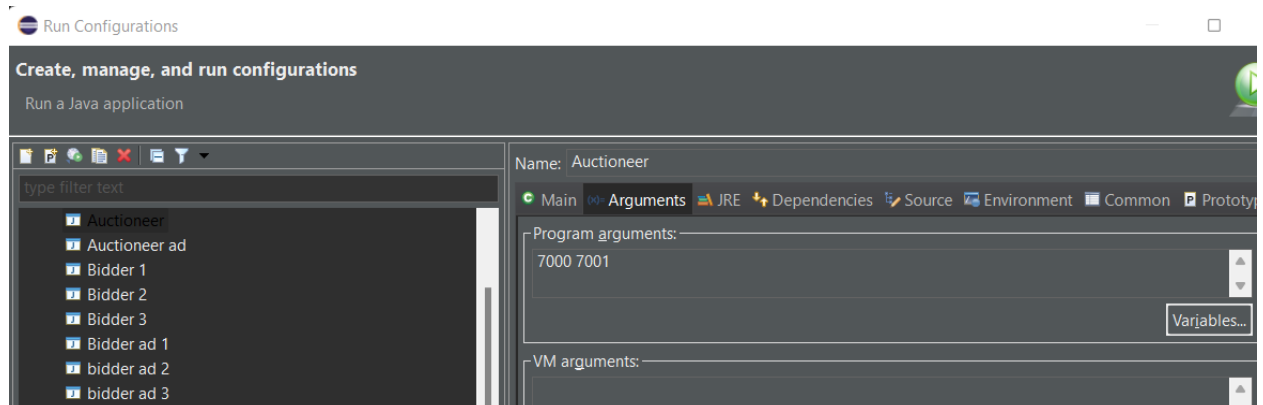
## Bidders basic solution output:

```
<terminated> Bidder 1 [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe (Dec 5, 2021, 11:02:30 AM – 11:02:36 AM)
node hostname is LAPTOP-OBTQ2TF0:LAPTOP-OBTQ2TF0/127.0.0.1
Bidder : 7001 of distributed lottry is active ....
Bidder : 7001 - forwarded token to 7002
```

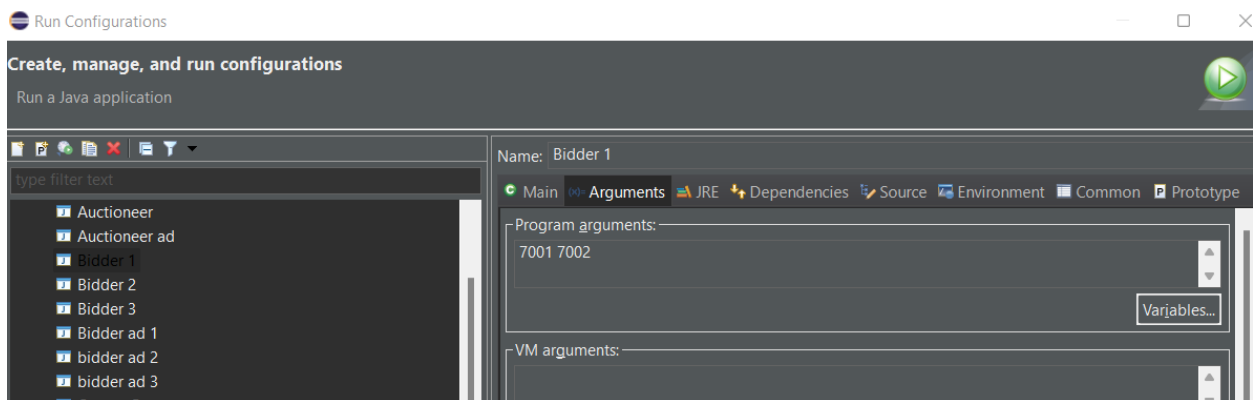
```
<terminated> Bidder 2 [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe (Dec 5, 2021, 11:02:33 AM – 11:02:37 AM)
node hostname is LAPTOP-OBTQ2TF0:LAPTOP-OBTQ2TF0/127.0.0.1
Bidder : 7002 of distributed lottry is active ....
Bidder : 7002 - forwarded token to 7003
```

```
<terminated> Bidder 3 [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe (Dec 5, 2021, 11:02:34 AM – 11:02:38 AM)
node hostname is LAPTOP-OBTQ2TF0:LAPTOP-OBTQ2TF0/127.0.0.1
Bidder : 7003 of distributed lottry is active ....
Bidder : 7003 - forwarded token to 7000
```

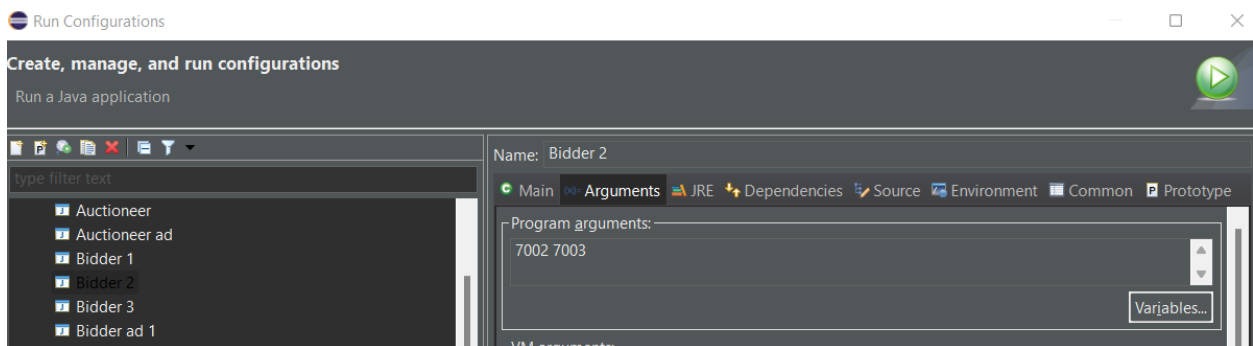
## Auctioneer Run Configuration Basic Solution:



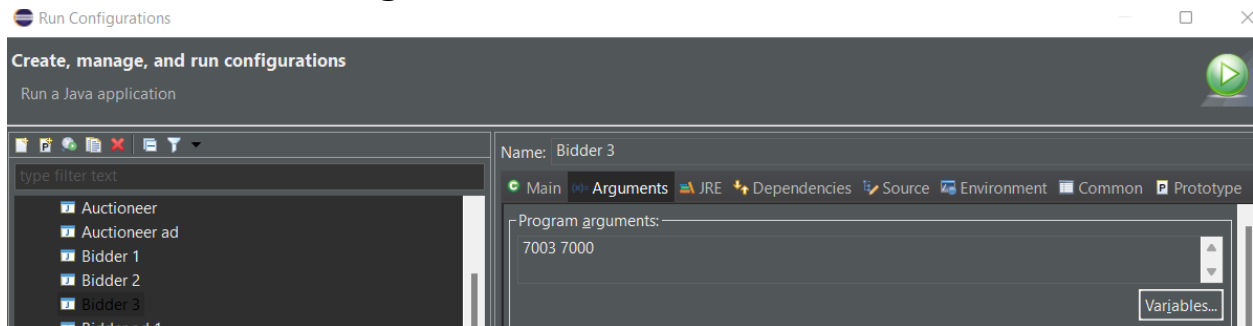
## Bidder 1 Run Configuration Basic Solution:



## Bidder 2 Run Configuration Basic Solution:



## Bidder 3 Run Configuration Basic Solution:



## Advanced Solution:

Auctioneer class:

For this part the only difference between the basic and the advanced version is that a for loop was placed for the program to go continuously until the designated number of rounds has been completed so that multiple bets can be taken, and a new parameter or argument was added in the main to record the number of rounds that has been given. In this case 3 rounds were given.

The loop completes once every time the token is received and the \_bid.txt is updated.

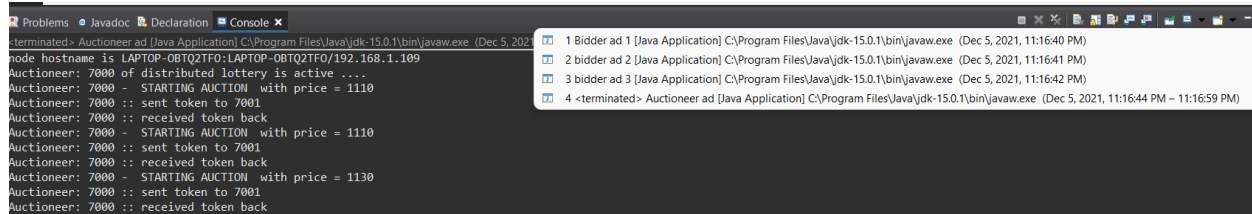
Bidder Class:

The only major difference here would be that the while(true) is used to make an infinite number of loops for the bidders to continue bidding each time they receive the token or if no bids are generated. Even when the rounds are completed then the bidders will continue to wait for the token.

Run configuration:

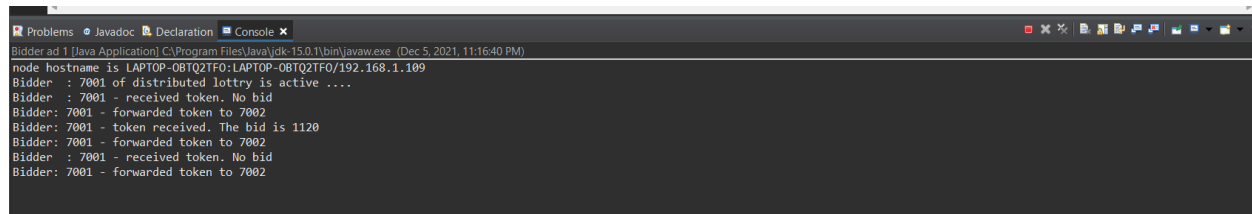
For the auctioneer the argument needed an extra parameter which was the number of rounds or loops the system must complete. The arguments for the bidders remain the same as in basic solution.

## Auctioneer Advanced solution output with display of bidders running:

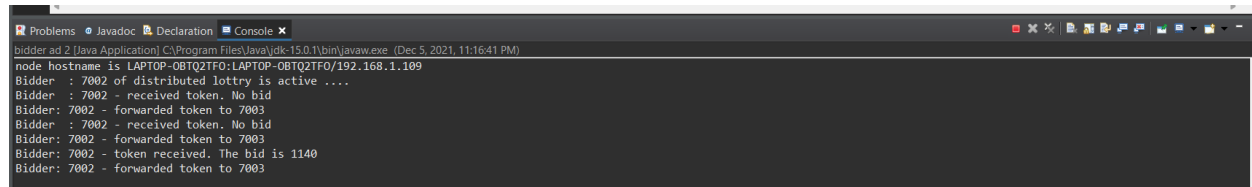


```
terminated> Auctioneer ad [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe (Dec 5, 2021, 11:16:40 PM)
node hostname is LAPTOP-OBTQ2TFO:LAPTOP-OBTQ2TFO/192.168.1.109
Auctioneer: 7000 of distributed lottery is active ....
Auctioneer: 7000 - STARTING AUCTION with price = 1110
Auctioneer: 7000 :: sent token to 7001
Auctioneer: 7000 :: received token back
Auctioneer: 7000 - STARTING AUCTION with price = 1110
Auctioneer: 7000 :: sent token to 7001
Auctioneer: 7000 :: received token back
Auctioneer: 7000 - STARTING AUCTION with price = 1130
Auctioneer: 7000 :: sent token to 7001
Auctioneer: 7000 :: received token back
```

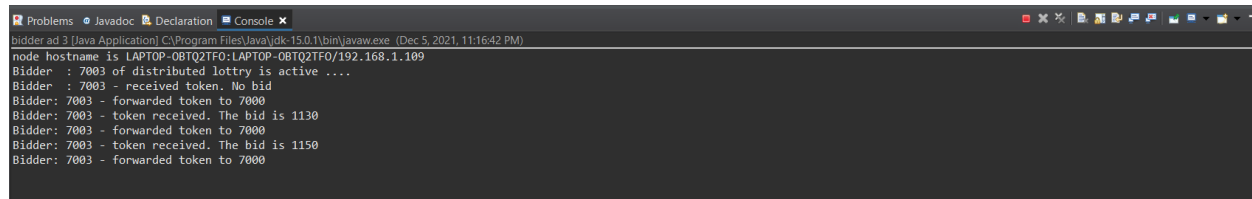
## Bidder Advanced solution output:



```
Bidder ad 1 [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe (Dec 5, 2021, 11:16:40 PM)
node hostname is LAPTOP-OBTQ2TFO:LAPTOP-OBTQ2TFO/192.168.1.109
Bidder : 7001 of distributed lottery is active ....
Bidder : 7001 - received token. No bid
Bidder: 7001 - forwarded token to 7002
Bidder: 7001 - token received. The bid is 1120
Bidder: 7001 - forwarded token to 7002
Bidder : 7001 - received token. No bid
Bidder: 7001 - forwarded token to 7002
```

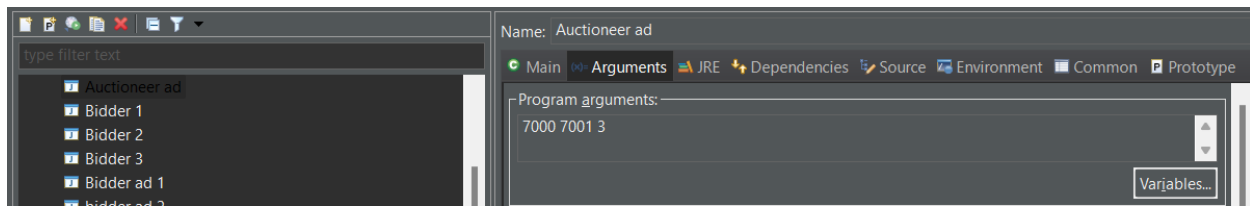


```
bidder ad 2 [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe (Dec 5, 2021, 11:16:41 PM)
node hostname is LAPTOP-OBTQ2TFO:LAPTOP-OBTQ2TFO/192.168.1.109
Bidder : 7002 of distributed lottery is active ....
Bidder : 7002 - received token. No bid
Bidder: 7002 - forwarded token to 7003
Bidder : 7002 - received token. No bid
Bidder: 7002 - forwarded token to 7003
Bidder: 7002 - token received. The bid is 1140
Bidder: 7002 - forwarded token to 7003
```



```
bidder ad 3 [Java Application] C:\Program Files\Java\jdk-15.0.1\bin\javaw.exe (Dec 5, 2021, 11:16:42 PM)
node hostname is LAPTOP-OBTQ2TFO:LAPTOP-OBTQ2TFO/192.168.1.109
Bidder : 7003 of distributed lottery is active ....
Bidder : 7003 - received token. No bid
Bidder: 7003 - forwarded token to 7000
Bidder: 7003 - token received. The bid is 1130
Bidder: 7003 - forwarded token to 7000
Bidder: 7003 - token received. The bid is 1150
Bidder: 7003 - forwarded token to 7000
```

## Auctioneer Advanced solution run configuration:



### Advanced Solution part 2 discussion:

If one of the nodes does crash, then the whole system is affected as the system is based on ring topology. So, even though the connections between the will still be active there won't be any data sharing as the token can't go beyond the crashed node. This crash could happen if one of the nodes is abruptly closed or if the connection between the nodes has failed to stay connected. If the node has crashed before sending the token then the data sharing (the bids) will stop at the crashed node, but if the crashed happened after token was sent from the crashed node then the data sharing will continue to happen for that round as the connections between the other nodes are still stable.

### Most relevant sections:

I would say that the most relevant sections would be where the `_bid.txt` is being updated as that is a critical section of the entire system where the information is being shared and updated. In both the auctioneer and bidder class.

The other most relevant section would be the out socket as that is where the token will be leaving for both classes and carrying the bid information with it.

### Things that were incomplete in the assignment:

The program doesn't display the final winner (highest bidder) of the auction.

Also, I had forgotten to add comments about the new parameter for the advanced solution in the main of auctioneer.

Functionally, I think that the program is as complete as it can be for the assignment.

## End of Report